Chinese Function Tag Labeling[⋆]

Weiwei Sun[a] and Zhifang Sui[b]

[a] Department of Computational Linguistics, Saarland University
German Research Center for Artificial Intelligence (DFKI)
Saarbrücken, D-66123, Germany
wsun@coli.uni-saarland.de
[b] Institute of Computational Linguistics, Peking University
Key Laboratory of Computational Linguistics, Ministry of Education, China
Beijing, 100871, China
szf@pku.edu.cn

Abstract. Function tag assignment has been studied for English and Spanish. In this paper, we address the question of assigning function tags to parsed sentences in Chinese. We show that good performance for Chinese function tagging can be achieved by using *labeling* method, extending the work of Blaheta (2004). In this method, the objects being modeled are syntax trees which require some mechanism to convert them into feature vectors. To encode structural information of the complex inputs, we propose a set of new features. Experimental results show that these new features lead to significant improvements.

Keywords: function tagging, function tag labeling

1 Introduction

In the Penn TreeBank, function tags appended to constituent labels are used to indicate additional syntactic or semantic information. Modern statistical parsers such as Collins and Charniak parsers ignore much of functional information, although the training corpora are annotated with this kind of additional information. Nevertheless, there is an increasing interest in enriching the output of parsers with function tags in the last few years. A number of algorithms have been proposed for English and Spanish. But little is known about how these algorithms may perform in many other languages. In this paper we address the question of assigning function tags to parsed sentences in Chinese. To the authors' knowledge, this is the first attempt to evaluate function tagging approaches on Chinese.

Two function tag assignment approaches have been presented in previous research: *parsing* approach and *labeling* approach. In this paper, we focus on the second one. We implement the same function tag labeling system which is proposed by (Blaheta, 2004). The realization of baseline features, such as category clusters, will be discussed. Developing features that capture the right kind of information is crucial to explore function tagging labeling. Although previous work has shown great promise, the features used in previous work have not fully exploited what a syntax tree provides. We propose some new features to convert syntax trees into feature vectors.

We evaluate on both hand-crafted and automatic parsing syntax trees to clarify the performance of models in Chinese function tag labeling. Our system[1] shows that good function tagging results

[⋆] The work was partially completed while the first author was at Peking University. The first author is funded both by German Academic Exchange Service (DAAD) and German Research Center for Artificial Intelligence (DFKI). This work is partially supported by NSFC Project 60873156, National Social Science Fund (China) 09BYY032, and the Project of Toshiba (China) Co., Ltd. R&D Center. We would like to thank Meng Wang and Zhengyu Niu for their helpful discussion.

[1] Source code is available at `http://code.google.com/p/chinesefunctiontagging/`

for Chinese can be achieved by using labeling method, achieving 94.47 F-measure on gold standard syntax trees and 81.63 F-measure on trees parsed by Charniak parser. Experimental results also show that our new features lead to fairly significant improvements over the baseline.

2 Function Tags in Chinese Penn TreeBank

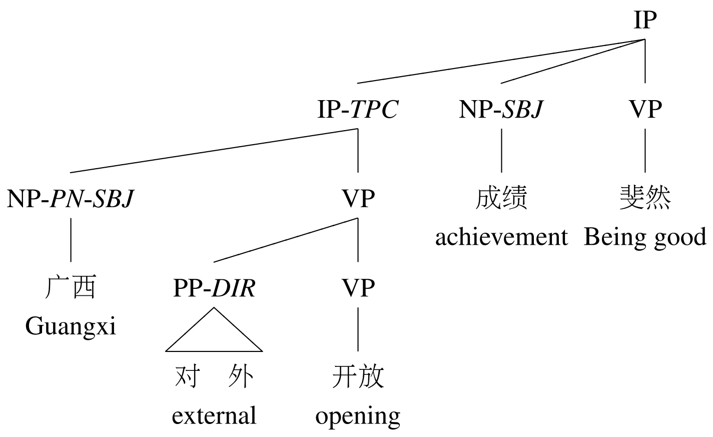

Figure 1: A sentence with function labels

In the Penn TreeBank, function tags appended to constituent labels are used to indicate additional syntactic or semantic information. Figure 1 shows a simplified tree representation with function labels in Chinese Penn TreeBank (CTB). For example, *SBJ* marks the surface subject *"广西/Guangxi"*; this noun phrase is also a proper noun and is thus assigned *PN*; the preposition phrase *"对外/external"* is tagged with a *DIR* label, which indicates the direction role of the predicate "开放/opening". Table 1 is the complete list of function labels and their brief explanation in CTB (Xue and Xia, 2000). Table 2 is the numbers of each function tag type in CTB 5.0.

Table 1: Chinese Penn TreeBank Function Tags

Syntactic Label		**Semantic Label**	
IO	indirect object	BNF	benefactive
OBJ	direct object	CND	condition
EXT	extent	DIR	direction
FOC	focus	IJ	interjective
PRD	predicate	LGS	logical subject
SBJ	surface subject	LOC	locative
TPC	topicalized	MNR	manner
Miscellaneous Label		PRP	purpose or reason
APP	appositive	TMP	temporal
HLN	headline	VOC	vocative
PN	proper nouns	**Clause Type**	
SHORT	short form	IMP	imperative
TTL	title	Q	question
WH	Wh-phrase	**Discrepancy Label**	
		ADV	adverbial

Current statistical parsers do not use functional information because performance of the parser usually decreases considerably, since a more complex task is being solved. Despite its importance, functional information, hence, is usually ignored in statistical parsing work. Nevertheless, there has been some work concentrating on the function tag assignment tasks. In this paper, we concentrate on this task in Chinese.

Table 2: Numbers of Function Tags. In CTB, empty category can also hold function tag. For each tag type, the first number is without empty category and the second one is within empty category.

Syntactic Label			**Semantic Label**		
IO	228	237	BNF	645	645
OBJ	43701	47410	CND	624	624
EXT	1485	1544	DIR	2724	2739
FOC	247	247	IJ*	28	29
PRD	6113	6117	LGS	398	399
SBJ	38995	65335	LOC	4684	4799
TPC	2291	3149	MNR	2424	2618
Miscellaneous Label			PRP	1109	1173
APP	6762	6762	TMP	8254	8475
HLN	1501	1501	VOC*	20	20
PN	28559	28581	**Clause Type**		
SHORT*	60	60	IMP*	49	49
TTL	887	891	Q	931	935
WH	832	832	**Discrepancy Type**		
			ADV	4400	4430

3 Method

3.1 Previous Work

There are two main kinds of function assignment methods, which we call *parsing* method and *labeling* method. Parsing methods integrate function tag assignment into the parsing process (Gabbard et al., 2006; Merlo and Musillo, 2005), whereas labeling approaches take syntactic parsing as pre-processing and label function tags or *NULL* tag (which indicates the given constituent does not represent any function tags) to each syntax tree node. (Blaheta, 2004; Jijkoun and de Rijke, 2004; Chrupała and van Genabith, 2006).

Gabbard et al. (2006) modify Collins parser' model 2 to allow it to produce function tags without decreasing the parsing performance. In the original model function tags is deleted after being used to identify and mark arguments. Collins parser use function tags as part of the heuristics for doing so. A following pre-processing step then deletes the orininal function tags. In Gabbard et al.'s approach, the parser retains the function tags after using them for argument identification, and therefore includes them in all the parameter classes. Merlo and Musillo (2005) extend a Simple Synchrony Network (SSN) parser to produce richer output annotated with function tags. The main idea of their modification is to split some part-of-speech tags into tags marked with semantic function labels. Their functional parser reaches state-of-the-art results both in parsing and in function tag assignment (Merlo and Musillo, 2005).

3.2 Labeling Method in This Paper

A parsing task combining function tags and general categories is more complex than simple parsing. A majority of categories are respectively divided into several correlated ones. For example, category *IP* in Figure 1 is divided into *IP-HLN* and *IP-TPC*. One obstacle to parsing methods is the sparse problem caused by this *sub-categorizing* process. This will be more severe for CTB since there are much less sentences than the English TreeBank.

In this paper, we implement a *labeling* method for Chinese function tagging, following (Blaheta, 2004). This approach takes function tagging as classification tasks. Given a syntax tree, our system extracts a variety of features to represent every non-terminal node; probability (or distance to the seperating hyperplane) for each possible function tag is then computed from the features. According to the labeling guideline of CTB (Xue and Xia, 2000), function tags can be divided into five categories. Table 1 shows the complete list of function labels. In English TreeBank, a constituent can be tagged with multiple tags, but never with two tags from the same category. In CTB, however, though a constituent cannot be labeled with two tags from each syntactic, semantic and clause category, it can be assigned more than one tag from miscellaneous labels. So we can take syntactic and semantic label tagging as two multi-category classification subtasks, and other function labels as binary classification subtasks. Table 2 shows the numbers of each function tag type. Tags *VOC*, *IJ*, *SHORT* and *IMP* are extremely sparse in CTB 5.0 (no more than 60 instances for each one). It is impossible to learn them for most machine learning algorithms, so we exclude them in our experiments. In summary, our system consists of several classifiers for:

- syntactic function tag labeling;
- semantic function tag labeling;
- other 7 binary classification subtasks.

For example, the syntactic function tag classifier may predict that 广西*(Guangxi)* in Figure 1 is a *SBJ* whereas the *proper noun* classifier recognizes that phrase as a *PN*. Other classifiers, such as *HLN* predictor, should assign *NULL* label to this phrase.

4 Features

4.1 Baseline Features

Our baseline system uses features introduced by Blaheta (2004): **category**, **cc-category**, **head**, **head POS**, **alt head**, **alt head POS**, **category clusters**.

- **Category** This is the syntactic category (NP, VP, IP, etc.) of the constituent.
- **cc-Category** If a candidate phrase is comprised of the conjunction of two or more *X*P, this feature is CC*X*P.
- **Head** To extract the syntactic head of a phrase, we use head rules described in (Sun and Jurafsky, 2004). This set of head rules are very popular in Chinese parsing research, such as in (Duan et al., 2007; Zhang and Clark, 2008).
- **Head Word POS** The part-of-speech of syntactic head.
- **Alt Head Word** Many kinds of function tags, such as temporals and locatives, occur as prepositional phrases in a sentence, and it is often the case that the head words of those phrase, which are always prepositions, are not very discriminative, for example, "在去年/in the last year", "在北京/in Beijing", both share the same head word "在", but the former is **TMP** whereas the latter is **LOC**. *Alt Head Word* feature is the head of the object of a prepositional phrase (and undefined for other sorts of constituents), which is designed to capture more information of prepositional phrases.
- **Alt Head Word POS** The part-of-speech of the alt head.
- **Category clusters** Blaheta manually created a number of category clusters on English. In our labeling system, we present a similar rule on Chinese which is summarized in Table 3.

4.2 New Word-based Features

To improve labeling performance, we propose many other kinds of features containing plentiful information. These features include:

Table 3: Category Clusters

C1: VCD, VCP, VNV, VP, VPT, VRD, VSB				
C2: DNP, DP, FW, NN, NP, PN				
C3: ADVP, DVP, MSP			C4: LCP, PP	
C5: CP, FRAG, IP			C6: CLP, QP	
C7: ADJP	C8: LST	C9: PP	C10: PRN	C11: UPC
C12: Other categories				

- **Boundary words** Some constituents tend to contain discriminative first and last words, as well as the words surrounding these constituents. We try to use them along with their POS tags, and these features include:
  1. the first word current phrase, its POS tag and word cluster (ifw, ifpos, ifc);
  2. the last word current phrase, its POS tag and word cluster (ilw, ilpos, ilc);
  3. one word before current phrase and its POS tag (ofw, ofpos);
  4. one word after current phrase and its POS tag (olw, olpos).
- **Combining features** We also combine some orignial features as new features, including:
  1. conjunction (ifw-ilw) of the first and last words;
  2. conjunction (ifpos-ilpos) of the POS tags of the first and last words;
  3. conjunction (ofw-olw) of the outside words;
  4. conjunction (ofpos-olpos) of the POS tags of the outside words.
- **Head words of children** These features include:
  1. Head word (fhw, lhw) of the first and last child;
  2. POS tags of head word (fhpos, lhpos) of the first and last child.
- **Length** (len) The number of words in current phrase.

4.3 New Structural Features

In this task, the objects being modeled are syntax trees which require some mechanism to convert them into feature vectors. Taking syntax trees as inputs, the classifiers should characterize structural properties of syntactic parses, and the design of features to represent syntactic structures requires research effort. We put forward a number of new features to encode the structural information:

- **Rewrite rule** Rewrite rules (rr, prr) expand current phrase and its parent. For *prr* feature, to distinguish current phrase node and its sisters, we make a symbol to locate the current node. For example, the *rr* feature for *IP-TPC* in Figure 1 is *NP→NP,(VP)*, and the *prr* feature is *IP→IP,NP,(VP)*.
- **Combining Categories** Conjunction (c-pc) of the categories of current phrase and its parent. Conjunction (c-pc-gpc) of the categories of current phrase and its parent and grandparent. Conjunction (lc-rc) of the categories of the two siblings. For example, the *lc-rc* feature of “成绩/achievement” in Figure 1 is *IP-VP*.
- **POS chain** The sequential containers (pos-c) of each word’s POS. Single character POS chain (spos-c): each POS in a POS chain is clustered to a category defined by its first character. For *IP-TPC*, these feature are **NR-P-NN-VV** and **N-P-N-V**.

Table 4: Effect of each feature on the function tag classification when added to the baseline.

Feature	Overall	Syntactic	Semantic
Baseline	86.33	94.95	75.82
+ifw	***87.70**	**95.25**	***78.24**
+ifpos	***87.82**	**95.15**	**77.22**
+ifc	***87.12**	**95.13**	**76.49**
+ilw	***87.63**	94.86	***81.87**
+ilpos	***87.24**	**95.15**	***78.48**
+ilc	***87.52**	94.89	***81.10**
+ofw	86.24	**95.06**	***76.17**
+ofpos	**86.43**	94.87	**76.98**
+olw	***86.84**	**95.07**	***78.55**
+olpos	**86.55**	**95.03**	**76.84**
+ifw-ilw	***87.88**	**95.03**	***81.26**
+ifpos-ilpos	***88.85**	**95.26**	***80.22**
+ofw-olw	**86.68**	**95.23**	**77.23**
+ofpos-olpos	**86.43**	**94.96**	**76.55**
+fhw	***89.24**	***95.33**	***78.23**
+fhpos	***88.89**	***95.41**	***77.92**
+lhw	**86.68**	94.94	***78.09**
+rhpos	**86.72**	94.92	***80.95**
+len	***87.07**	**94.98**	**76.82**
+rr	***90.40**	***95.39**	***78.79**
+prr	***88.42**	***96.87**	***78.38**
+c-pc	**86.56**	**94.97**	**75.95**
+c-pc-gpc	**86.81**	**95.08**	**76.92**
+lc-rc	**86.51**	94.89	***77.36**
+pos-c	***88.80**	94.83	***80.32**
+spos-c	***87.74**	94.86	***78.02**
+cct-c	***89.86**	94.73	***79.00**
+cct-w	***87.78**	**94.98**	***78.34**
+htr	***89.06**	**95.06**	***77.69**

- **Head Trace** (htr) The sequential container of the head down upon the phrase. For example, the head word of *IP-IPC* is “开放/opening”; therefore this feature of *IP-TPC* is $IP{\downarrow}VP{\downarrow}VP{\downarrow}VV$. This feature is very similar to etree feature in TAG grammar (Liu and Sarkar, 2007).
- **C-commander thread of the head** C-commander[2] thread features, raised by (Sun et al., 2008), are sequential containers of constituents which C-command the head word of the constituent. We design two C-commander threads:
  1. all items in the thread are categories of the C-commanders (cct-c);
  2. using the word content to occupy the head position (cct-w).

  For instance, in Figure 1, the noun phrase “广西/Guangxi” and the preposition phrase “对外/opening” are two left c-commanders of the head “开放/opening”, so the *cct-w* feature for *IP-TPC* is *NP*←*PP*←开放.

[2] C-command is a concept in X-bar syntax theory. Assuming α and β are two nodes in a syntax tree: α C-commands β means every parent of α is ancestor of β.

5 Experiments and Analysis

5.1 Experimental Setting

CTB contains comprehensive functional information, and CTB 5.0 is used in our experiments. There are 890 files and 18807 sentences in CTB 5.0. In our experiments, we divided these files into three parts:

- files from chtb_021 to chtb_1100 are used as training set;
- files form chtb_1101 to chtb_1130 as development set;
- files form chtb_001 to chtb_020, and chtb_1131 to chtb_1151 as test set.

In addition, all empty categories are excluded. There are 14853, 1853, 1974 sentences in each data set.

5.2 Feature Performance

Table 4 shows the performance of baseline, and the effect each feature has on three tasks: 1) all labels, 2) syntactic labels, and 3) semantic labels, when added individually to the baseline. These results are based on syntactic features extracted from hand-crafted TreeBank parses. Here, maximum entropy model is used for classification. On all the system improvements, we perform a binominal test of significance at p=0.05, and all significant improvements are marked with a *. From this table, we can see that the most effective features are those so called structural features.

5.3 Classifier Performance

Table 5: F-measures for different classifiers.

	Syntactic		Semantic	
	devel.	test	devel.	test
SNoW	96.92	96.60	82.28	84.19
MaxEnt	97.15	96.53	85.74	86.72
SVM	97.44	96.94	85.46	87.19

We experimented with three popular machine learning algorithms: Support Vector Machine classifier (SVM) (Vapnik, 1998), Maximum Entropy classifier (ME) (Berger et al., 1996), and Sparse Network of Winnows (SNoW) (Roth, 1998). For each algorithm we use the same set of features. In terms of SVM, we used TinySVM[3]. All SVM classifiers were realized with default parameters. One-Vs-All strategy is used to solve multi-class classification problem. For ME model, we use maxent[4]. For SNoW model, we use UIUC SNoW toolkit[5]. In training, SNoW's default parameters are used with the exception of the separator thickness 1.5, the use of average weight vector, and 5 training cycles.

Table 5 shows the classification performance of different classifiers, both on test and development data set. We can see that these three algorithms show a very similar performance on syntactic labels, while SVM outperforms both, scoring 96.94 on F-measure. SVM and ME model perform similarly on semantic labels, and work much better than SNoW.

5.4 Function Tag Labeling Performance

Table 6 shows the overall tagging performance with all features. The syntactic tags which are useful for recognizing arguments get a very high performance. Comparison with baseline performance (in Table 4) indicates that our new features significantly improve this task. Table 7 shows

[3] `http://chasen.org/~taku/software/TinySVM/`
[4] `http://homepages.inf.ed.ac.uk/s0450736/maxent\_{}toolkit.html`
[5] `http://l2r.cs.uiuc.edu/~danr/snow.html`

Table 6: Overall performance on gold parses.

	P(%)	R(%)	$F_{\beta=1}$
Overall	95.68	93.30	94.47
Syntactic labels	97.05	96.82	96.94
Semantic labels	88.57	85.85	87.19

Table 7: Detailed performance on gold parses.

	P(%)	R(%)	$F_{\beta=1}$
EXT	82.50	72.79	77.34
FOC	70.00	70.00	70.00
IO	100.00	58.62	73.91
OBJ	97.55	98.98	98.26
PRD	98.04	97.56	97.80
SBJ	97.46	98.07	97.76
TPC	84.55	60.43	70.48
BNF	83.33	97.22	89.74
CND	71.70	58.46	64.41
DIR	84.42	88.26	86.30
LGS	75.76	83.33	79.37
LOC	92.06	81.36	86.38
MNR	81.93	89.45	85.53
PRP	92.86	79.59	85.71
TMP	91.59	89.24	90.40
ADV	87.39	72.22	79.09
PN	97.94	92.37	95.08
HLN	91.93	83.15	87.32
APP	95.72	88.54	91.99
Q	76.11	55.84	64.42
WH	76.03	76.67	76.35

the detailed labeling performance for all function labels. Some sparse tags cannot be accurately identified, such as *Q*. The recall, in general, performs worse than precision. This is mainly for that the negative samples (syntactic nodes are not assigned any function label) is much more than the positive sample.

5.5 Using Automatic Parses

The results in former experiments are based on the use of hand-crafted parses. In practical use, of course, automatic parses will not be as accurate. To gauge the tagging performance in realistic situation, in this section, we report experiments on function tag labeling with automatic parsing information. Charniak (Charniak and Johnson, 2005) parser, which is ported to Chinese, are used to produce full parses. The parser is re-trained using the same training and development data in function tagging. Table 8 summarizes the parsing performance of Charniak parser. Table 9 shows the parsing performance and the function tag labeling performance (evaluation metric was described in (Blaheta and Charniak, 2000)).

6 Conclusion

Function tag assignment has been studied for English and Spanish. In this paper, we address the question of assigning function tags to parsed sentences in Chinese. We describe a Chinese

Table 8: Parsing performance.

	LP(%)	LR(%)	$F_{\beta=1}$
Overall	70.83	75.68	73.12
Len<=40	76.35	79.07	77.68

Table 9: Function tag Labeling performance on automatic parses.

	P(%)	R(%)	$F_{\beta=1}$
Overall	81.49	81.76	81.63
Syntactic	86.01	86.10	86.05
Semantic	80.39	77.42	78.88

function tag labeling system and show that good function tagging performance for Chinese can be achieved. A variety of new features are proposed to improve the system. As a process based on syntax trees, structural properties of full syntactic parses should be encoded as classifier's features. To do this, we introduce a number of structural features to convert syntax structures into flat feature representations. Experiments show that structural information of the inputs are extremely important for function tag labeling, and that our new features characterizing structural information yield significant improvements.

References

Berger, Adam L., Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1).

Blaheta, Don and Eugene Charniak. 2000. Assigning function tags to parsed text. In *Proceedings of the First Conference on North American Chapter of the Association for Computational Linguistics*.

Blaheta, Don. 2004. *Function Tagging*. Ph.D. thesis, Brown University, Providence, RI, USA. Adviser-Eugene Charniak.

Charniak, Eugene and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Chrupała, Grzegorz and Josef van Genabith. 2006. Using machine-learning to assign function labels to parser output for Spanish. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*.

Duan, Xiangyu, Jun Zhao, and Bo Xu. 2007. Probabilistic models for action-based Chinese dependency parsing. In *ECML '07: Proceedings of the 18th European conference on Machine Learning*, pages 559–566, Berlin, Heidelberg. Springer-Verlag.

Gabbard, Ryan, Mitchell Marcus, and Seth Kulick. 2006. Fully parsing the penn treebank. In *Proceedings of Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*.

Jijkoun, Valentin and Maarten de Rijke. 2004. Enriching the output of a parser using memory-based learning. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*.

Liu, Yudong and Anoop Sarkar. 2007. Experimental evaluation of LTAG-based features for semantic role labeling. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 590–599, Prague, Czech Republic, June. Association for Computational Linguistics.

Merlo, Paola and Gabriele Musillo. 2005. Accurate function parsing. In *HLT '05: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*.

Roth, Dan. 1998. Learning to resolve natural language ambiguities: a unified approach. In *AAAI '98/IAAI '98: Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*.

Sun, Honglin and Daniel Jurafsky. 2004. Shallow semantic parsing of Chinese. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*.

Sun, Weiwei, Zhifang Sui, and Haifeng Wang. 2008. Prediction of maximal projection for semantic role labeling. In *Proceedings of the 22nd International Conference on Computational Linguistics*.

Vapnik, Vladimir N. 1998. *Statistical Learning Theory*. Wiley-Interscience, September.

Xue, Nianwen and Fei Xia, 2000. *The Bracketing Guidelines for the Penn Chinese Treebank*.

Zhang, Yue and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, Honolulu, Hawaii, October. Association for Computational Linguistics.